This is the post peer-review accepted manuscript of:

Fabio Viola, Ariane Stolfi, Alessia Milo, Miguel Ceriani, Mathieu Barthet,and György Fazekas. 2018. Playsound.space: enhancing a live music per-formance tool with semantic recommendations. In1st International Work-shop on Semantic Applications for Audio and Music (SAAM '18), October 9,2018, Monterey, CA, USA.ACM, New York, NY, USA

The published version is available online at https://doi.org/10.1145/3243907.3243908

Playsound.space: enhancing a live performance tool with semantic recommendations

Fabio Viola
ARCES, University of Bologna
Bologna, Italy
fabio.viola@unibo.it

Ariane Stolfi
University of São Paulo
São Paulo, Brazil
a.stolfi@qmul.ac.uk

Alessia Milo
Queen Mary University of London
London, United Kingdom
a.milo@qmul.ac.uk

Miguel Ceriani
Queen Mary University of London
London, United Kingdom
m.ceriani@qmul.ac.uk

Mathieu Barthet
Queen Mary University of London
London, United Kingdom
m.barthet@qmul.ac.uk

György Fazekas
Queen Mary University of London
London, United Kingdom
g.fazekas@qmul.ac.uk

ABSTRACT

Playsound is a simple and intuitive tool for collaborative music composition based on sounds from Freesound, an online repository of diverse audio content with Creative Commons licenses. In this paper, we present an approach based on Semantic Web technologies to provide recommendations to Playsound users. A Semantic Web of Things architecture is outlined, showing loosely coupled, independent software agents interoperating by means of a semantic publish/subscribe platform and a set of ontologies to describe agents, audio contents, input/output of audio analytics tools and recommendations. Preliminary tests confirm that the designed architecture adapts well to environments where services can be discovered and seamlessly orchestrated on the fly, resulting in a dynamic workflow.

CCS CONCEPTS

• **Information systems → Web Ontology Language (OWL)**; *Collaborative and social computing systems and tools*; Service buses;

KEYWORDS

Recommendations, Semantic Web, Web of Things, Ontologies

ACM Reference Format:
Fabio Viola, Ariane Stolfi, Alessia Milo, Miguel Ceriani, Mathieu Barthet, and György Fazekas. 2018. Playsound.space: enhancing a live performance tool with semantic recommendations. In *Proceedings of Workshop on Semantic Applications for Audio and Music (SAAM 2018)*, Editor John, Editor Jane, and Editor Jim (Eds.). ACM, New York, NY, USA, Article n, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Playsound.space[1] has been developed as a tool for live performance exploiting the Freesound[2] database, based on semantic queries and visual feedback provided by spectrograms. The design process and first user evaluations have been presented in a previous paper [28]. Subsequently, a collaborative environment for people to play with has been realized by introducing more complex tools for audio transformations and resources. Among these extensions, we present in this paper a recommendation system for *Playsound* that provides suggestions that users can use as inspirations or as new samples for their artworks.

Recommendations are nowadays ubiquitous, being intensively employed by all of the most diffuse web applications; they are in fact used by social networks (to suggest possible friendships or interesting topics), entertainment companies (to advice movies, albums or songs based on user's history), online retailers (to suggest items similar to the one we are looking at) just to name a few. The problem of recommending contents is not new in literature and has been extensively studied for more than two decades [8, 22]. In this paper, the focus is on the way content for recommendations is accessed, analyzed and provided, rather than on advances on recommendations algorithms.

Playsound's recommendations can be considered as the result of a collaborative step where the leading actors are not the users, but a set of web agents cooperating in a *Semantic Web of Things* (SWoT) ecosystem pivoting a semantic publish/subscribe middleware. The functionalities offered by the agents consists in search for content on online sound databases (i.e. Europeana[3], Freesound and Jamendo[4]), analysis of audio files and, of course, recommendation. The *Web of Things* (WoT) paradigm [20] allows to dynamically discover services offering such functionalities, invoke and combine them (i.e. orchestrate) through a detailed interface presented in the so-called *Thing Description*. The use of semantics allows to foster interoperability, achieving a common representation of data according to a set of ontologies. This approach overcomes the problem of interacting with online databases that use conflicting custom data representations. Furthermore, it is intrinsically extensible to include new services on-the-fly. In fact, in the presented system, agents are automatically discovered and this influences the workflow that adapts to the context.

The developed recommendation system is a content-based one, that exploits tags of the audio files selected by the users to perform a tag-based search on multiple online datasets (i.e. Europeana,

[1] http://playsound.space
[2] https://freesound.org

[3] https://www.europeana.eu
[4] https://www.jamendo.com

SAAM 2018, Oct. 2018, Monterey, California USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Freesound and Jamendo). The research is then refined by carrying out an audio analysis to check the similarity between the retrieved audio signals and the one selected by the user. Being Playsound a collaborative environment, the current recommendation system can be further evolved toward a content-based, collaborative one [5] providing recommendations based on all the users involved in a session.

The rest of the paper is organized as follows: Section 2 proposes an overview of the state of the art of dynamic workflow organization and SWoT applications. Section 3 introduces the background and the motivation behind this work. In Section 4 the recommendation system is presented: first the foreseen workflow is described (Section 4.1), then the software architecture is introduced (Section 4.2) and eventually the semantics of the operations is described (Section 4.3). Two practical examples (Section 5) demonstrates the presented work, while in Section 6, conclusions are drawn.

2 RELATED WORK

In this paper, we present a semantic recommendation system for a collaborative composition tool. Our aim is to explore how Semantic Web and *Internet of Things* (IoT) technologies and paradigms may be used to fulfill complex tasks by combining from time to time the available elementary pieces. As highlighted by Barker and Van Hermert in [6], loosely coupled *service-oriented architectures* (SOA) are the key to flexibility and scalability. Furthermore using standards enhances the interoperability. In our approach we adopt a slight variation of the SOA paradigm, the SWoT, to realize a multi-agent system that operates with a dynamic workflow. The latter is based on discovering available functionalities in the network and orchestrating them to achieve the final scope.

Similarities with the work presented in this paper can be found in [9], where the authors propose an actor-oriented workflow modeling. In our case, actors are the (virtual) devices, the so-called *web things* (WT). Web things provide services through their actions. Actions have input and output, just like actors, and they are all documented by the thing description, a machine-readable document. Dynamic workflows that adapt to the context can also be found in [19]. There, the authors present a context-aware environment exploiting semantic processing and situational method engineering to automatically adapt workflows to the current situation. With respect to the cited works, we propose an approach where the workflow adapts to the available web things by performing a discovery on the central context broker based on the actions provided by each web thing. Both the discovery and invocation of the desired actions are performed through SPARQL (respectively with queries and updates). Furthermore, results provided by each web things can be timely received thanks to SPARQL subscriptions.

While the WoT is emerging as the latest evolution of the IoT toward the use of web standards, the SWoT is more focused on the adoption of semantic technologies: notwithstanding the advantages of using such technologies, their adoption is still not so broad and the research in this area is still in its early stages. A pioneristic work in the area of the SWoT is the one by Pfisterer et al. [24]. The W3C Web of Things Working and Interest Groups are trying to standardize the way devices can be discovered and accessed from the Web, and a preliminary vocabulary has been proposed by W3C members in [13]. Serena et al. proposed in [27] one of the first ontologies to map devices in the (Semantic) Web of Things to achieve a semantic discovery of the available things, based on the W3C terminology. In the present paper we leverage on these two related work using a more detailed SWoT ontology suitable for discovering, but also accessing devices through a semantic publish/subscribe broker in a fully semantic environment. A similar approach was first applied in [4], where authors focused on elderly and impaired people assistance through an indoor positioning monitoring application. In this paper, we apply the same approach to web services, by abstracting the concept of web thing.

3 BACKGROUND

Playsound is a tool for live performances, supporting free improvisation by providing a rich palette of sounds accessed through Freesound APIs. The need to provide recommendations of audio files in Playsound, according to the ones selected by the user, motivates our work. On the Web, several repositories hosting Creative Commons audio files exist and recommendations would highly benefit from a wider set of contents. Unfortunately, each audio repository adopts a custom data representation formalism. Achieving a full interoperability among these is possible employing Semantic Web data representation mechansims and platforms. In this Section we introduce the main application as well as the platform adopted as the interoperability enabler and the project in which this work is framed.

3.1 Playsound

Playsound was originally born to support artists during live performances. It can be used for musical practices such as free improvisation, experimental music production and soundscape composition. To achieve this scope, Playsound exploits Freesound APIs [2]. These APIs also provide access to spectrograms, an effective way to visually represent a sound when dealing with non-musical samples [28]. From the architectural point of view, Playsound is a client/server application relying on NodeJS (server-side) and Angular (client-side). Users only need a web browser supporting HTML5 and the Web Audio APIs to run Playsound. The architecture is shown in Fig. 1, while a screenshot of the user interface is reported in Fig. 2. The right-hand side section is the *search panel*. There, typing a keyword in the proper field triggers a call to Freesound API. Results are shown in the search panel, where every audio file is represented with its spectrogram. The left-hand side section is the *play panel*: every file selected by the user among the search results is added to the play panel.

3.2 SEPA

Sharing information organized in a semantic way according to a set of ontologies, requires a proper infrastructure. An RDF datastore is usually employed to store such information and make it available through a standard interface relying on SPARQL query and update languages. In dynamic environments, where information need to be timely dispatched to the requesting clients, a SPARQL endpoint may not be enough. The *SPARQL Event Processing Architecture* (SEPA) [25] addresses this limitation by implementing the publish/-subscribe paradigm [15] on top of a standard SPARQL Endpoint.

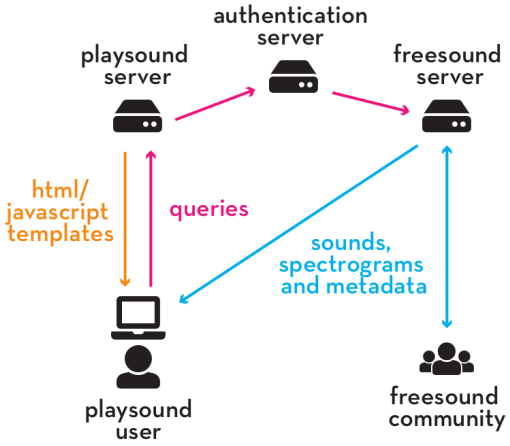

Figure 1: Playsound software architecture

Figure 2: Playsound User Interface: play panel (left) and search panel (right)

SEPA is based on the experience matured on the Smart-M3 platform [14, 26] and provides the SPARQL Subscribe Language that allows a client to specify a subgraph of interest with a high granularity and receive a notification when an event (i.e. a modification of the knowledge base) occurs. SEPA implements a delta-notification mechanism, so the notification only contains added and removed bindings, and not the whole set of information matching the issued subscription, resulting in compact messages to reduce the impact on performance. In such an architecture, every client, also known as *knowledge processor* (KP) is said to be a *producer* when involved in modifications to the knowledge base, a *consumer* when it performs read-only operations and an *aggregator* if performs updates of the knowledge base to react to a given event (Fig. 3).

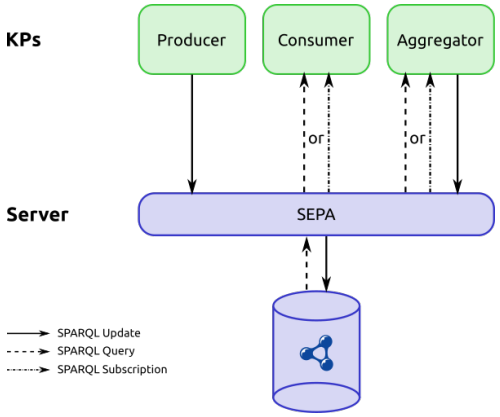

Figure 3: Interaction between SEPA clients (KPs) and server

3.3 AudioCommons

Audio Commons Initiative[5] [17] is a European research project aimed at bridging the gap between the amount of Creative Commons audio content published every day and the content that is really exploited by professional creative industries in their work.

In the Audio Commons vision, content creators will expose their content in the Audio Commons Ecosystem through a number of content providers that host and publish it according to a common metadata specification. This role is played by the Audio Commons ontology [12] that will be the main interoperability enabler.

4 RECOMMENDATIONS THROUGH SEMANTICS

In this paper, we demonstrate how Playsound may benefit from the adoption of Semantic Web technologies. The Semantic Web, introduced by Tim Berners Lee [7], aims at transforming the web from a repository of human-readable information, to a world wide set of also machine-understandable data. This is achieved through the introduction of the Semantic Web stack that is composed by several standards allowing (among others) to univocally identify resources (IRI), represent data (RDF), provide a vocabulary of the relevant terms (RDFS and OWL) and retrieve/push information (SPARQL). Semantic technologies are often seen as an interoperability enabler for heterogeneous agents. This is the key aspect that motivates their adoption in the Playsound application. Despite being a tool oriented at live performance, Playsound also provides recommendations based on the samples selected by the users for their composition. In this way, the performers can not only enhance their track replacing selected samples with those suggested by the application, but also be inspired by them. Recommendations come from several different online databases (i.e. Freesound, Jamendo and Europeana).

4.1 Workflow

The basic Playsound workflow consists of the iteration of the following simple steps:

[5] https://www.audiocommons.org/

(1) Perform a keyword-based search for samples on Freesound (i.e. from the search panel introduced in Section 3).
(2) Select sounds and add them to the play panel on the left hand side where a new player object is created.

Whenever a file is added to the left panel, a request for recommendations is issued by the client. The system exploits a content-based recommendation system based on the tags of the audio files that time after time are added to the left panel. A filtering stage allows to refine the research by means of similarity measures. These steps are the results of the cooperation of several very simple entities detailed in the following Section.

4.2 Software Architecture

Playsound's recommendation system has been implemented according to a semantic-enhanced version of the WoT paradigm [20]. Software agents are then modeled as *virtual* devices offering services (i.e. actions) and generating events. Actions, events and properties of every web thing compose the so-called thing description, an ID card of every service. All the web things implementing the recommendation system are independent software agents, that coexist in a semantic ecosystem. The latter is a layered system (depicted in Fig. 4) where the web things stand on top of a set of ontologies. The ground layer is represented by an RDF store underlying a SPARQL Event Processing Architecture that provides a semantic publish-subscribe mechanism for real-time notifications of events. That said, the software architecture may be presented by traversing the semantic ecosystem layers from top to bottom.

4.2.1 Layer 1: The web things. The first layer is represented by the web things composing the recommendation system:

- **AudioQuery Server WT** – This web thing implements the software agent providing a semantic description of the Playsound server. It is responsible for generating recommendation requests to the proper web thing.
- **Europeana WT**, **Freesound WT**, **Jamendo WT** – These are the web things providing search mechanisms for Europeana, Freesound and Jamendo respectively. They are software agents acting as semantic bridges to the original APIs providing a search mechanism that returns data represented through the Audio Commons ontology. Multiple instances of these web things may be running at the same time, providing a redundancy that can be useful to address different needs (e.g. if they run with different settings) or to efficiently serve clients in case of high number of requests.
- **Sonic Annotator WT** – A software agent offering Sonic Annotator [11] as a service. By invoking its action, it is possible to perform feature extraction and audio annotation based on the VAMP plugins hosted on the operating system running the WT. Multiple instances of the Sonic Annotator web thing may coexist in the same ecosystem: users (e.g. other web things) may invoke the instance they prefer according to a given policy (in our example we always select the instance with the lowest computational load according to the property exposed by the web thing).
- **Recommender WT** – A service that performs orchestration of requests by discovering and invoking Europeana, Jamendo and Freesound actions (if available) to search for audio files and then discovering and invoking Sonic Annotator's action to compute similarity measures.

Exploiting a publish/subscribe platform (detailed later on), web things are loosely coupled software agents, decoupled in space, time and synchronization [15]. In fact, web things do not need to know each other (thanks to a dynamic discovery mechanism) and neither to share the same hosting server. Furthermore, not all the web things need to be active at the same time, since the recommendation system may work with the available ones (if any) or publish a request that will be satisfied by the first web thing to connect. Last, but not least, the entire communication among web things is asynchronous.

4.2.2 Layer 2: The ontologies. The Playsound recommendation subsystem exploits, through these web things, five ontologies:

- **Semantic Web of Things ontology** – developed at the University of Bologna with the aim to foster interoperability among heterogeneous devices in the IoT. It is based on a previous work by Serena et al. [27] and according to the current W3C's Web of Things terminology [13]. The SWoT ontology is used to provide a semantic description of all the web things in terms of actions, properties and events (i.e. the thing description) and to allow the interaction with them.
- **Audio Commons ontology** [12] – designed to have common data model to search and interact with audio resources, as required by the European Research Project "Audio Commons". The Audio Commons ontology generalizes the Music Ontology [16] and extends the FRBR ontology. All the web things offering a search action exploit the Audio Commons ontology to map the results in a uniform way (i.e. performing then a mapping from the service specific API to the common ontology).
- **Audio Features ontology** – The Audio Features ontology (AF) [3] allows sharing content-derived information about musical recordings and is used by Sonic Annotator to represent all the extracted audio features.
- **Vamp Plugins Ontology** – it describes the Vamp API used to invoke vamp plugin system for audio analysis. More specifically, the Vamp Plugins ontology defines all that is necessary to define a *transform*, the input of a plugin (being a transform the couple *plugin-input settings*) [10].
- **Recommendation ontology** – based on the Similarity ontology, the DCMI Metadata Terms, the Association ontology and the Ordered List ontology and designed to provide basic concepts and properties for describing recommendations [1].

4.2.3 Layer 3 and 4: SPARQL Event Processing Architecture and RDF datastore. If the above-mentioned ontologies allow to represent data in a uniform and unequivocal way, a further software component is needed to host this information. An RDF store (also known as triple store) plays the role of a semantic information broker among the web things. RDF stores usually expose a SPARQL interface through which retrieving and updating data according to the SPARQL query and update languages. However, as already mentioned in Section 3.2, dynamic environments need proper mechanisms to timely issue notification of changes in the knowledge base. This motivates the adoption of SEPA [25] that wraps the RDF

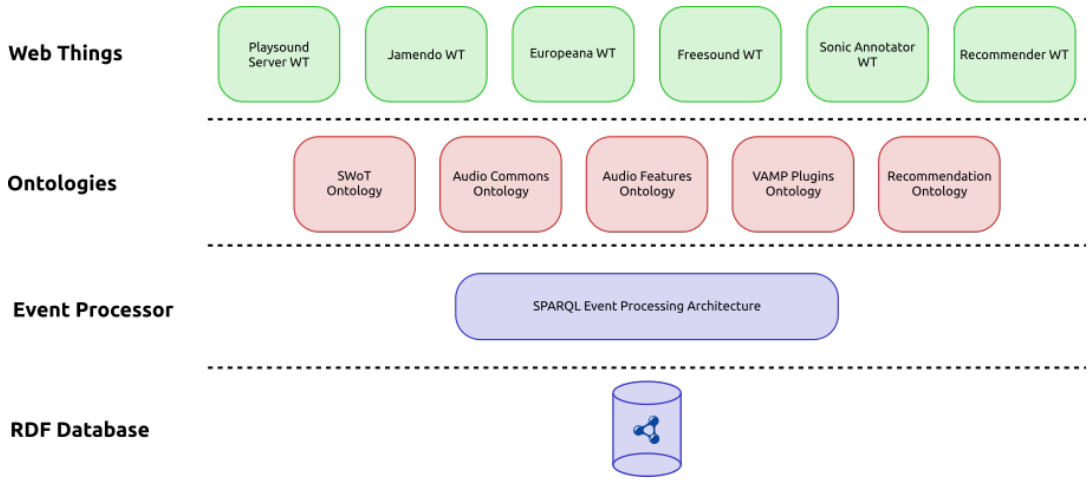

Figure 4: Software architecture for the recommendation system

store providing a content-based publish-subscribe functionality based on the SPARQL query language.

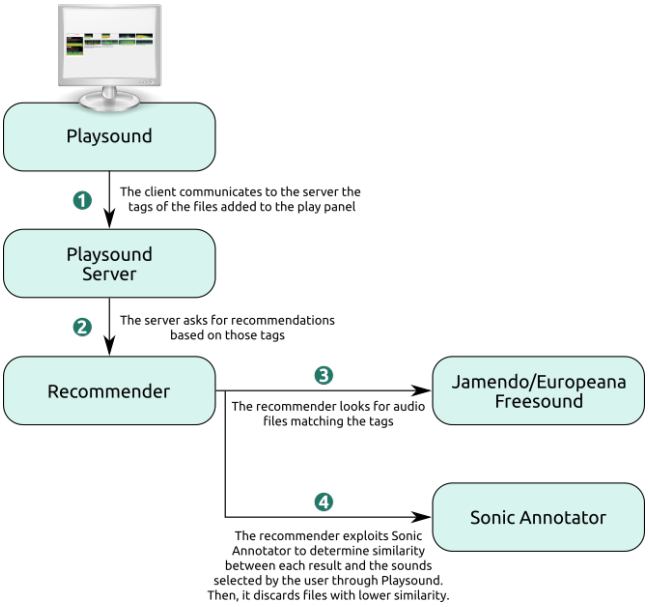

Figure 5: Playsound's recommendations sub-system

4.3 Semantics of Recommendations

A new request for recommendation issued by Playsound creates a named graph that is incrementally modified by the web things in the ecosystem. In fact, when the recommendation web thing invokes Europeana, Freesound and Jamendo, they put in the graph all the results (if any) of their tag-based search. Then, when Sonic Annotator WT is requested to analyze the audio files, the calculated audio features are attached to the proper resources in the named graph. Finally, the recommender is able to execute a SPARQL query that selects the audio files with the higher similarity and provide results to the client.

The following algorithm shows how the Recommender WT operates when it gets a new request for recommendations for file `audioFile`. The second parameter contains the tags bound to the audio file, while the third one is the named graph where results should be put.

```
1:  function RECOMMEND(audioFile, tags, graph)
2:
3:      ▷ Look for web things with search capabilities
4:      swt ← sepa.discover(action="search")
5:
6:      ▷ Iterate over all the web things in swt
7:      for wt ∈ swt do
8:          ▷ Issue a non-blocking search request
9:          sepa.requestAction(wt, search, tags, graph)
10:     end for
11:
12:     ▷ Look for devices providing audio analytics services
13:     awt ← sepa.discover(action="execVampPlugin")
14:     if awt ≠ ∅ then
15:         ▷ Select one of these web things
16:         awt1 = awt.pop()
17:         ▷ Request the computation of the spectral centroid (SC)
18:         sepa.requestAction(awt1, spectral_centroid, graph)
19:     end if
20:
21:     ▷ Produce recommendations
22:     sepa.doUpdate(Recommend r with SC similar to audioFile)
23:
24: end function
```

Lines 4 and 13 carry out two discovery tasks (i.e. by means of the SPARQL query language) by looking for web things respectively able to search on audio databases and to analyze audio files. The loop starting at line 7 issues a series of search requests to all the available web things to look for files with the given tags. This policy, suitable for the simple test scenario, can be strengthen by adding constraints that limit the number of web things to activate. The

results of every search operation are expressed according to the Audio Commons ontology and put into the named graph specified by the argument `graph`. On line 16, a web thing providing audio analytics services is selected: once again, the adopted selection method consists in choosing the one with the lowest computational workload. Line 18 is used to request the computation of the spectral centroid on all the audio files in the named graph. On line 19, a SPARQL update is issued to the SEPA instance to request the creation of an instance of the class `rec:Recommendation` containing a property `rec:recommended_in` for all the audio files with a spectral centroid similar to the provided one. It is worth mentioning that the method named `requestAction` is a SPARQL update used to put into SEPA an action request compliant with SWoT ontology: all the web things get notified about the pertaining action requests through SPARQL subscriptions.

4.3.1 Europeana / Freesound / Jamendo. The three web things described in this Section are very similar, since they invoke the proper web APIs to search for audio files matching a set of tags and build a semantic representation of the results. In fact, all the services (i.e. Europeana, Freesound and Jamendo) provide a response according to a custom format. Then, in order to achieve a common representation of all the results, the related web things exploit the Audio Commons ontology [12]. SPARQL-generate [21] was used to map all the results from the custom JSON documents to the given ontology. To clarify this process, we provide an example based on the reply provided by Jamendo for a simple search request. For the sake of brevity, only one of the results is shown and the fields not used by the recommendation system were removed.

```
{
  "results": [
    {
      "id": "1157664",
      "name": "Alain (instrumental)",
      "audiodownload": "https://mp3d.jamendo..",
          "shorturl": "http://jamen.do/t/1157664",
      ...
    }
  ]
}
```

In order to map the results according to the Audio Commons ontology, an instance of the class `ac:AudioClip` must be created for every result, and should be linked through the `ac:available_as` property to an instance of the class `ac:AudioFile`. The title of the audio clip should then be linked to the clip through the property `dc:title`. Then, the SPARQL generate query needed to perform this mapping is the following:

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX ac: <http://audiocommons.org/ns/audio..>
PREFIX iter: <http://w3id.org/sparql-generate..>
PREFIX fn: <http://w3id.org/sparql-generate/fn/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf..>
GENERATE {
  ?audioClip rdf:type ac:AudioClip .
  ?audioClip ac:available_as ?audioFile .
  ?audioClip dc:title ?title .
  ?audioFile rdf:type ac:AudioFile
}
SOURCE <link_to_jamendo_api> AS ?source
ITERATOR
  iter:JSONPath(?source,"$..results[*]") AS ?res
WHERE {
  BIND(fn:JSONPath(?res, ".id" ) AS ?id)
  BIND(IRI(fn:JSONPath(?res, "shorturl")) AS
    ?audioClip)
  BIND(IRI(fn:JSONPath(?res, "audiodownload")) AS
    ?audioFile)
  BIND(fn:JSONPath(?res, "name") AS ?title)
}
```

where the query encloses the call to Jamendo API (with the keyword `SOURCE`). JSONPath allows to navigate the reply message, binding every field to a variable that is subsequently used by the `GENERATE` section to produce the output triples. An example of the output is:

```
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ac: <http://audiocommons.org/ns/audio..> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf..> .
@prefix fn: <http://w3id.org/sparql-generate/fn/> .
@prefix iter: <http://w3id.org/sparql-generate..> .

<http://jamen.do/t/1157664> a ac:AudioFile .

<https://mp3d.jamendo..> a ac:AudioClip ;
    ac:available_as <http://jamen.do/t/1157664> ;
    dc:title "Alain (instrumental)" .
```

4.3.2 Sonic Annotator. The tag-based research performed by the web things described in the previous Section may not be as accurate to suggest audio clips that are really similar to the one selected by the user. For this reason a very simple filtering stage is involved in the recommendation system. Sonic Annotator, through its web thing agent, calculates audio features for each audio clip suggested by the above mentioned services. The similarity is computed by comparing the spectral centroids of the signals, a weighted mean of the frequencies present in the signal representing its brightness [18]. The brightness is one of the most important features for an audio signal, so it is a good candidate for estimating similarity [23]. The spectral centroid could be also replaced by or combined with other audio features without altering the nature of the system: in fact, multiple vamp plugins are available through the Sonic Annotator WT, but an exhaustive discussion of the similarity measures is out of the scope of the paper. In order to calculate the spectral centroid, the VAMP plugin `linearcentroid` provided by the VAMP SDK is used.

The input of Sonic Annotator is the so-called transform, i.e. a VAMP plugin and a set of parameters. Sonic Annotator is fed with an N3 file containing the transform expressed according to the VAMP Plugins ontology [10] (a transform is then an instance of the class `vamp:Transform`). The Audio Features ontology [3] is instead used to express the output of the analysis.

5 EXAMPLES

A first basic test of this proof of concept is presented in this Section. Searching "`8 bit`" on Playsound, forty results are presented to the user on the first page (due to the pagination applied by Playsound). Among the results, we add to the play panel the file named "`8-Bit explosion`". This operation causes Playsound to asks for recommendations. The first step, consists in looking for available web things offering a search service for audio databases. In this example, we suppose that three web things, one for Europeana, one for Freesound and one for Jamendo are available. The first search mechanism is tag-based and the tags attached to the selected file are: `computerized`, `wav`, `bit`, `8` and `8bit`. Jamendo and Freesound return a list of five elements each (the size is a configurable parameter of the web things), Europeana only two (due to the license of the content). The second step, the similarity analysis based on the spectral centroid, allows to reject two results from Jamendo's list, three from Freesound's one and all of the results coming from Europeana. The following list shows a summary of the results of the first step, reporting the source, the name of the audio file, and an asterisk if the file is filtered out in the second step.

- **Jamendo**
  - Ode to Zork – by: Octabitron*
  - Bouncy Chips – by: Melhadf*
  - The warp repeater theme – by: Dementialcore
  - TELEPORTER – by: DANJYON KIMURA
  - danza terrestre – by: Kamarina SOUND MATHINE
- **Freesound**
  - Zerothru9.wav*
  - boxes.wav*
  - Pattern01.wav*
  - SequenceText1.wav
  - Craxy.wav*
- **Europeana**
  - Viaduct Westrandweg Halfweg*
  - Mikail Ivanovic Glinka: "Russlan e Ludmilla - Ouverture"*

As expected, shutting down Jamendo web thing, the recommendation service still provides suggestion (i.e. only the file named "`Pattern01.wav`"), while on the other hand closing Freesound web thing only three songs provided by Jamendo will be suggested to the user. Of course, shutting down Europeana's service nothing changes.

With the same set of running web things, a second test has been performed. Searching for "husky howl", the tender nickname attributed to Led Zeppelin's Robert Plant singing, we selected the audio sample "Igor-13B.wav". The set of tags bound to this file are: `moan`, `growl`, `malamute`, `dog`, `Wolf`, `husky`, `bark` and `howl`. Both Freesound and Jamendo returned five results for this set of keywords. Two less for Europeana. The analysis carried out by Sonic Annotator, allowed to reject two suggestions from Freesound and Europeana (considered too different from the original file), and one from Jamendo.

- **Jamendo**
  - Curious Day With Rufus Hot Sauce, Op 192 – by: Edward Schaffer
  - The night drives the wolf – by: DJ Mircomix*
  - The Wolf (Acoustic) – by: Nemo Wilson
  - Reaction 7 - Cool Dog – by: Reaction 7
  - Alain (instrumental) – by: FilsTool
- **Europeana**
  - Dog barking and birds
  - Grey Wolf', 'Woodland*
  - Love Is A Dog*
- **Freesound**
  - Igor-13C.aif*
  - Igor- I wan't dinner.wav
  - Igor Dinner Anticipation.wav
  - Igor17B-a drink of water.aif*
  - Igor16B-big talk.aif*

6 CONCLUSION

In this paper, a recommendation system for the collaborative music composition tool Playsound has been presented. Adopting a Semantic Web of Things approach, multiple independent agents semantically described in a central broker provide search and analysis of audio files and cooperate to achieve the common task of recommending audio contents. The system is able to adapt to the context by dynamically discovering the available agents and selecting the audio analysis service with the lowest computational load (if any). A detailed evaluation of this experimental work will be carried out as a future work. Furthermore, the integration of another (type of) web thing providing a multi-lingual translation and synonym research for tags is one of the improvements expected for the next release. Another enhancement will be represented by incremental recommendations, that will allow to suggest audio files not only based on the last file added to the play panel, but also on the previous items in the same artwork.

REFERENCES

[1] 2010. The Recommendation Ontology 0.3. http://smiy.sourceforge.net/rec/spec/recommendationontology.html

[2] Vincent Akkermans, Frederic Font Corbera, Jordi Funollet, Bram de Jong, Gerard Roma Trepat, Stelios Togias, and Xavier Serra. 2011. Freesound 2: An improved platform for sharing audio clips. *ISMIR Conference Proceedings* (2011). https://repositori.upf.edu/handle/10230/22726

[3] Alo Allik, György Fazekas, and Mark B Sandler. 2016. An Ontology for Audio Features.. In *ISMIR*. 73–79.

[4] F. Antoniazzi, G. Paolini, L. Roffia, D. Masotti, A. Costanzo, and T. S. Cinotti. 2017. A web of things approach for indoor position monitoring of elderly and impaired people. In *2017 21st Conference of Open Innovations Association (FRUCT)*. 51–56. https://doi.org/10.23919/FRUCT.2017.8250164

[5] Marko Balabanović and Yoav Shoham. 1997. Fab: content-based, collaborative recommendation. *Commun. ACM* 40, 3 (1997), 66–72.

[6] Adam Barker and Jano Van Hemert. 2007. Scientific workflow: a survey and research directions. In *International Conference on Parallel Processing and Applied Mathematics*. Springer, 746–753.

[7] Tim Berners-Lee, James Hendler, and Ora Lassila. 2001. The semantic web. *Scientific american* 284, 5 (2001), 34–43.

[8] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. Recommender systems survey. *Knowledge-based systems* 46 (2013), 109–132.

[9] Shawn Bowers and Bertram Ludäscher. 2005. Actor-oriented design of scientific workflows. In *International Conference on Conceptual Modeling*. Springer, 369–384.

[10] C Cannam. 2009. The Vamp Plugin Ontology.

[11] Chris Cannam, Michael O. Jewell, Mark Sandler, Christophe Rhodes, and Mark d'Inverno. 2010. Linked Data And You: Bringing music research software into the Semantic Web. *Journal of New Music Research* 39, 4 (2010), 313–325.

[12] Miguel Ceriani, Gyorgy Fazekas, Johan Pauwels, Mathieu Barthet, and Mark Sandler. [n. d.]. Deliverable D2.3 Final Ontology Specification.

[13] Victor Charpenay, Sebastian Käbisch, and Harald Kosch. 2016. Introducing Thing Descriptions and Interactions: An Ontology for the Web of Things.. In *SR+ SWIT@*

ISWC. 55–66.

[14] Alfredo D'Elia, Fabio Viola, Luca Roffia, Paolo Azzoni, and Tullio Salmon Cinotti. 2017. Enabling Interoperability in the Internet of Things: A OSGi Semantic Information Broker Implementation. *International Journal on Semantic Web and Information Systems (IJSWIS)* 13, 1 (2017), 147–167.

[15] Patrick Th Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. 2003. The many faces of publish/subscribe. *ACM computing surveys (CSUR)* 35, 2 (2003), 114–131.

[16] György Fazekas and Mark B Sandler. 2012. Knowledge Representation Issues in Audio-Related Metadata Model Design. In *Audio Engineering Society Convention 133*. Audio Engineering Society.

[17] Frederic Font and Xavier Serra. [n. d.]. THE AUDIO COMMONS INITIATIVE. ([n. d.]).

[18] Martin Gasser, Arthur Flexer, and Thomas Grill. 2011. On computing morphological similarity of audio signals. In *Proceedings of the 8th Sound and Music Computing Conference, Padova, Italy*.

[19] Gregor Grambow, Roy Oberhauser, and Manfred Reichert. 2010. Semantic workflow adaption in support of workflow diversity. (2010).

[20] Dominique Guinard and V Trifa. 2016. *Building the web of things*.

[21] Maxime Lefrançois, Antoine Zimmermann, and Noorani Bakerally. 2017. A SPARQL extension for generating RDF from heterogeneous formats. In *European Semantic Web Conference*. Springer, 35–50.

[22] Deuk Hee Park, Hyea Kyeong Kim, Il Young Choi, and Jae Kyeong Kim. 2012. A literature review and classification of recommender systems research. *Expert Systems with Applications* 39, 11 (2012), 10059–10072.

[23] Jouni Paulus and Anssi Klapuri. 2002. Measuring the similarity of Rhythmic Patterns.. In *ISMIR*.

[24] Dennis Pfisterer, Kay Romer, Daniel Bimschas, Oliver Kleine, Richard Mietz, Cuong Truong, Henning Hasemann, Alexander Kröller, Max Pagel, Manfred Hauswirth, et al. 2011. SPITFIRE: toward a semantic web of things. *IEEE Communications Magazine* 49, 11 (2011), 40–48.

[25] Luca Roffia, Paolo Azzoni, Cristiano Aguzzi, Fabio Viola, Francesco Antoniazzi, and Tullio Salmon Cinotti. 2018. Dynamic Linked Data: A SPARQL Event Processing Architecture. *Future Internet* 10, 4 (2018), 36.

[26] Luca Roffia, Francesco Morandi, Jussi Kiljander, Alfredo D'Elia, Fabio Vergari, Fabio Viola, Luciano Bononi, and Tullio Salmon Cinotti. 2016. A semantic publish-subscribe architecture for the Internet of Things. *IEEE Internet of Things Journal* 3, 6 (2016), 1274–1296.

[27] Fernando Serena, María Poveda-Villalón, and Raúl García-Castro. 2017. Semantic Discovery in the Web of Things. In *International Conference on Web Engineering*. Springer, 19–31.

[28] Ariane Stolfi, Miguel Ceriani, Luca Turchet, and Mathieu Barthet. 2018. Playsound.space: Inclusive Free Music Improvisations Using Audio Commons. In *Proc. Nime*.